Profiling of Symmetric-Encryption Algorithms for a Novel Biomedical-Implant Architecture

Christos Strydis
christos@ce.et.tudelft.nl

Di Zhu
D.Zhu@student.tudelft.nl

Georgi N. Gaydadjiev
georgi@ce.et.tudelft.nl

Computer Engineering Laboratory, Electrical Engineering Dept.
Delft University of Technology
Postbus 5031, 2600 GA, Delft
The Netherlands

ABSTRACT

Starting with the implantable pacemaker, microelectronic implants have been around for more than 50 years. A plethora of commercial and research-oriented devices have been developed so far for a wide range of biomedical applications. In view of an envisioned expanding implant market in the years to come, our ongoing research work is focusing on the specification and design of a novel biomedical microprocessor core, carefully tailored to a large subset of existing and future biomedical applications. Towards this end, we have taken steps in identifying various tasks commonly required by such applications and profiling their behavior and requirements. One such task is decryption of incoming commands to an implant and encryption of outgoing (telemetered) biological data. Secure bidirectional information relaying in implants has been largely overlooked so far although protection of personal (biological) data is very crucial. In this context, we evaluate a large number of symmetric (block) ciphers in terms of various metrics: average and peak power consumption, total energy budget, encryption rate and efficiency, program-code size and security level. For our study we use XTREM, a performance and power simulator for Intel's XScale embedded processor. Findings indicate the best-performing ciphers across most metrics to be MISTY1 (scores high in 5 out of 6 imposed metrics), IDEA and RC6 (both present in 4 out of 6 metrics). Further profiling of MISTY1 indicates a clear dominance of load/store, move and logic-operation instructions which gives us explicit directions for designing the architecture of our novel processor.

Categories and Subject Descriptors

I.6.6 [**Simulation and modeling**]: [Simulation Output Analysis]; C.3 [**Computer Systems Organization**]: Special-purpose and application-based systems—*Real-time and embedded systems*; E.3 [**Data**]: Data Encryption—*Standards*

CF'08, May 5–7, 2008, Ischia, Italy.
Copyright 2008 ACM 978-1-60558-077-7/08/05 ...$5.00.

General Terms

Security, Performance

Keywords

Implantable devices, ultra-low power, symmetric encryption, microarchitectural profiling

1. INTRODUCTION

Microelectronics design has shifted in recent years to synthesizing low-power systems. A major vehicle towards this trend has been the radical shift, through enabling technology, to portable devices such as mobile phones and laptop computers. A field of science that has adhered to strict low-power constraints since its infancy is biomedical microelectronic implants and has been around for more than 50 years. Perhaps the most popular instance of such devices is the implantable pacemaker which, apart from saving lives, has acted as a catalyst on the general public closed-mindedness against biomedical implants. Indicative of the penetration and impact pacemakers have achieved is the fact that, in Europe alone, a total number of 299,705 implanted devices have been registered over the year 2003 (source: European Society of Cardiology [12]).

With the pacemaker being the flagship, biomedical implants are now being designed for a large, and constantly increasing, range of applications. These applications are primarily grouped into two main categories: physiological-parameter monitoring (for diagnostic purposes) and stimulation (actuation, in general) [27]. Instances of the former are devices measuring body temperature [33], blood pressure [13], blood-glucose concentration [25], gastric pressure [28], tissue bio-impedance [22] and more. In the latter category belong implantable pacemakers [5, 16] and implantable intracardiac defibrillators (ICDs) [31], various functional electrical stimulators for paralyzed limbs [26], for bladder control [23], for blurred cornea in the eye [24] and more pathoses.

In a world where clinical healthcare costs are increasing and population is aging, implant applications are expected to boom even further in the years to come. A future where people are moving around performing their everyday tasks while tiny implants are monitoring or assisting their body is maybe not so far. Implants are expected to be under the direct or indirect control of their hosts. Commands will be given to them to adjust their operation and biological data will be casually telemetered from them to logging stations at home or in a treating-physician's office for evaluation and diagnosis. This is already taking place with modern pacemaker devices. In this context, an aspect of microelectronic implants which has been largely overlooked so far is encryption of inbound and outbound traffic. Data needs to be relayed securely for protecting the personal biological information of patients. Conversely, commands to the implants need to be authenticated, protected and verified to originate from legitimate operators so as to avoid unwanted, malicious or even fatal in-vivo manipulation of the devices.

Our long-term work focuses on designing a novel, minimalistic, low-power processor suitable for a large subset of biomedical applications as the ones mentioned above. We are currently defining the architecture of such a digital processor. One of the examined benchmark suites, and a very important one as previously discussed, is encryption. In the current paper, we profile - through detailed simulations - a large set of popular encryption algorithms in terms of power consumption, energy expenditure, encryption rate and program-code size. We, then, select the ones with the best performance for the targeted application domain and investigate their respective instruction mixes in order to gain insight on the most suitable instructions for inclusion in our targeted architecture. The profiling platform, benchmark suite and assumptions made in our study are detailed in the following sections. Concisely, the contributions of this work are:

- To identify encryption algorithms which achieve the lowest average power consumption throughout their execution;
- To identify encryption algorithms with the overall lowest total energy budget for encryption various plaintext sizes;
- To identify encryption algorithms which encrypt various plaintext sizes at the highest possible rate or, at least, at a rate fast enough to satisfy the required sampling rate of the (biological) plaintext data;
- To identify the instruction mixes and frequencies of the best scoring encryption algorithms for including in the architecture of a new processor suitable for a variety of biomedical implants.

The rest of the paper is organized as follows: section 2 gives an overview of related works in the field and their goals and underlines the novel results presented in the current document. Section 3 outlines the framework onto which this profiling study has been built, based on the characteristics of biomedical implantable devices. Section 4 provides the details of our selected profiling testbed of encryption algorithms. Section 5 contains, in detail, the findings of this work. Overall conclusions and future work are discussed in section 6.

2. RELATED WORK

Much effort has already been spent on the profiling of encryption algorithms - especially in this context - in the field of wireless sensor networks (WSNs). Law et al. [19] have evaluated various block ciphers on a MSP430F149 core by Texas Instruments. Their focus is WSN applications and they evaluate their included ciphers in terms of security level, operation mode, computational effort and memory requirements. Energy figures are drawn implicitly through the number of execution cycles. They propose best cipher candidates for different combinations of available system memory and desired security level.

Luo et al. [21] evaluate block and stream ciphers for WSN-nodes in terms of memory requirements and execution time. Chang et al. [8] attempt energy measurements on RC5, DES and AES running on both the Ember and the CrossBow sensor nodes. Testing various plaintext sizes, they measure the energy costs of encryption, hashing and wireless transmission of data and assess the reduction in the lifetime of sensor nodes employing encryption.

Venugopalan et al. [29] evaluate the computational requirements of various stream/block ciphers and hash functions across a wide range of platforms. Based on their findings on the chosen platforms, they attempt to derive a multivariant model which allows the interpolation of performance for other, unevaluated architectures.

Grossschadl et al. [15] have used Sim-Panalyzer [3] to evaluate lightweight versions of RC6, RIJNDAEL, SERPENT, TWOFISH and XTEA in terms of performance, power and memory requirements. Their results indicate that carefully optimized versions of RC6 and RIJNDAEL can preserve their high performance while meeting tight code-size constraints. They also discuss the impact of key expansion and different modes of operation on the overall performance and energy consumption.

In this paper we are not attempting to present a detailed comparison of various encryption algorithms in terms of performance, power etc.. Evaluating them against various metrics and singling the best ones out is the first step. As a second step - and the focus of our ongoing work - we analyze the instruction mix and frequency of the best ciphers and draw directions for the microarchitectural design of a processor suitable for biomedical implants. The work presented here is novel in that it targets a different class of low-power devices with particular idiosyncrasies. To the best of our knowledge, no similar effort has been reported so far in explicitly provisioning an implant processor with data encryption.

3. CHARACTERISTICS OF THE PROFILED SYSTEMS

The biomedical-implant field calls for particular design requirements and constraints. This, combined with the fact that we do not attempt here a detailed comparison between different cryptographic schemes, has led to the following assumptions and parameters for our profiling experiment.

We have chosen to profile only symmetric encryption algorithms for two main reasons. Asymmetric schemes have been extensively investigated in the past and, due to their complexity, have been found to have computational and memory requirements that are prohibitively high for low-power embedded devices [11, 18]. Lately, there has been considerable work in the field, especially in WSNs, showing that carefully optimized software or hardware implementations of existing asymmetric algorithms may be viable for resource-constrained devices [14, 30]. Nonetheless, our choice is also based on the typical application scenario of targeted implants: data and especially command exchange with the implant does not happen particularly often, e.g once a day.

test name	size (B)	samples (#)	duration (sec)	sample rate (Sml/sec)	sample rate (KB/sec)
Blood Pressure (BP)	1404	141	0,282	500	4,86
Blood Pressure (BP)	12798	1198	2,396	500	5,22

Table 1: Biomedical workloads used for profiling.

The reasons for that primarily are the desired autonomous, unattended operation of such devices as well as the disproportionally large energy costs incurred when wirelessly transmitting data in-vivo. In effect, even if a combination of asymmetric and symmetric-key encryption is assumed for secure authentication and actual-data exchange, respectively, authentication is not expected to occur so often in an implant operation. It is, thus, not our primary concern for this profiling study which is focused on the most commonly executed task, i.e. the symmetrically-encrypted data exchange.

In the context of a typical application scenario, as discussed previously, (outbound) data telemetry takes place a lot more often than (inbound) command reception in implants. In effect, we are focused here on the encryption part of the profiled algorithms. Furthermore, due to their symmetric nature, most of these algorithms have the same computational requirements for both encryption and decryption.

Operation mode is the way for encrypting a message longer than the block size of an algorithm. In this work we only consider the Electronic CodeBook (ECB) mode. It has been shown that different operation modes (e.g. CBC, CFB, OFB) incur different fault-tolerance levels with regard to information loss due to transmitted-packet loss but incur the same energy penalty [19]. Since in this work we are not investigating the efficiency of different modes in terms of information integrity but, rather, profile ciphers based on their power signatures, ECB is sufficient.

The nature of the input datasets (i.e. plaintexts) does not affect the behavior of the studied encryption algorithms except for their size. For other algorithms, such as biological-data compression, the dataset nature does impact performance. However, in the interest of completeness, we have tested the algorithms against one biological workload provided from the BIOPAC (R) Student Lab PRO v3.7 Software. The BP workload represents continuous blood-pressure readouts. Our prior extensive study [27] on biomedical implants has revealed that typical data-memory sizes inside the implants range from 1 KB to 10 KB. Therefore, BP workloads of both sizes (1 KB and 10 KB, roughly) have been profiled. The specifics can be seen in Table 1.

4. EXPERIMENTAL SETUP

4.1 The simulator

The profiling of symmetric ciphers has been based on XTREM [10], a modified version of SimpleScalar [4, 7]. The XTREM simulator is a cycle-accurate, microarchitectural, power- and performance- functional simulator for the Intel XScale core. It models the effective switching node capacitance of various functional units inside the core, following a similar modeling methodology to the one found in Wattch [6]. XTREM has been selected for its straight-forward functionality but mostly for its high precision in modeling the performance and power of the Intel XScale core [17]. More precisely, it exhibits an average performance error of 6.5% and an even smaller average power error of 4% [9].

feature	value
ISA	32-bit ARMv5TE-compatibility, 8 DSP instructions
Pipeline depth	7/8-stage (depending on instruction), super-pipelined
Datapath width	32-bit
RF size	16 registers
Issue policy	in-order
Instr. window	single-instruction
I-Cache	32KB 32-way set-assoc. (1-cc hit/170-cc miss lat.)
D-Cache	32KB 32-way set-assoc. (1-cc hit/170-cc miss lat.)
TLB	32-entry fully-assoc.
BTB	128-entry direct-mapped
Branch Pred.	2-bit Bimodal
Write Buffer	8-entry
Fill Buffer	8-entry
Mem. bus width	4-byte
INT/FP ALUs	4/4
DSP co-proc.	40-bit, low-power, variable-lat. MAC
Clock freq.	2 MHz (typ. 200 MHz)
Oper. voltage	1.5 Volt
Implem. tech.	0.18 μm

Table 2: XScale architecture details.

The main XScale characteristics are summarized in Table 2. Many of the architectural features have been integrated into XTREM. Thumb instructions and special memory-page attributes are not supported but they do not affect simulation results since they are not used by our benchmarked applications. XTREM allows monitoring of 14 different functional units of the Intel XScale core: Instruction Decoder (DEC), Branch-Target Buffer (BTB), Fill Buffer (FB), Write Buffer (WB), Pend Buffer (PB), Register File (REG), Instruction Cache (I$), Data Cache (D$), Arithmetic-Logic Unit (ALU), Shift Unit (SHF), Multiplier Accumulator (MAC), Internal Memory Bus (MEM), Memory Manager (MM) and Clock (CLK).

Although XScale (and, thus, XTREM) is a low-power processor with aggressive power-management features, we are well-aware that it is not suitable for biomedical implants in terms of power consumption. However, our selection has been based on availability and on the crucial fact that XTREM models actual hardware with very high accuracy. Moreover, in this study we focus on the relative behavior and performance of different encryption algorithms rather than on absolute figures. We are interested in the differences observed across the various ciphers and resulting trends are highly probable to stay the same in our envisioned biomedical processor.

Last, clock frequency has been lowered to closer resemble realistic implantable systems. Other parameters have not been tampered with since it is not certain that the simulator will scale properly. For instance, instruction and data TLBs have not been disabled, the operating voltage or the memory latencies have not been altered.

encryption algorithm	block size (bits)	key size (bits)	Rounds (#)
3WAY [1]	96	96	11
BLOWFISH [1]	128	128	16
DES [1]	64	56	16
GOST [1]	64	256	32
IDEA [1]	64	128	8.5
LOKI91 [2]	64	64	16
RC5 [1]	64	128	12
SKIPJACK [2]	64	80	32
XXTEA [32]	64	128	32
MISTY1 [19]	64	128	8
RC6 [19]	128	128	20
TWOFISH [19]	128	128	16
RIJNDAEL [19]	128	128	12

Table 3: Benchmark suite of symmetric ciphers.

4.2 The encryption algorithms

When putting together our benchmark suite of ciphers, we have made an effort to include sources adhering to the following characteristics:

i. large range of symmetric encryption techniques and styles, from high-performing to compact flavors;

ii. mature, optimized, well-documented implementation code base;

iii. various algorithm complexities;

iv. suitability: the XTREM simulator can only handle C and Java sources. Furthermore, in its current version it does not support simulating an OS on top of the simulated hardware, thus prohibiting the use of encryption sources - such as the excellent bzip2 algorithm - that require multithreading support or other high-level features; and

v. availability: all collected benchmarks comprise utterly free, published or free under the GNU General Public License sources, readily available to the research community.

The implementation of a given encryption algorithm plays as crucial a role for the performance and behavior of the algorithm as its underlying structure. While adhering to the above characteristics, in order to offer the best possible fairness in our selection process, we have attempted to include algorithms built with the same implementation philosophy (e.g. algorithm suite implemented by the same author(s)) and/or algorithms being top representatives in their category. Table 3 summarizes our selected benchmark suite. An implementation of the original DES algorithm, although considered not secure any longer, has been included in our benchmark suite as a reference point for the rest of the considered ciphers.

5. PROFILING ANALYSIS

5.1 Power consumption

We start our profiling study by, first, examining how the selected ciphers perform in terms of power consumption since this is a crucial attribute of energy-constrained devices as implants. Overall and per-component average power consumption is depicted in Fig. 1 for all 13 ciphers and for the two BP plaintext sizes 1KB and 10 KB.

Across all ciphers we can readily see in the figure that the memory-manager unit (MM) is the most power-hungry component of the processor with a rough 69% fraction of overall power consumed. The MM unit is activated each time the core is stalled because of a main-memory instruction or data access. Next follow the ALU consuming roughly 18%, the clock structure (CLK) consuming 5% and the instruction-cache (I$) consuming 3.5% of the overall power, on average. Compared with other types of workloads, e.g. data compression, encryption is more computationally intensive (i.e. many arithmetic and logic operations), thus the high consumption of the ALU is not surprising. Further, encryption is typically data- rather than control-dominated, with few instruction branches, placing high demands on linear instruction fetch. That is why the instruction-cache consumes on average more power than other memory units, e.g. the data-cache or the BTB. Last, the clock structure is known throughout digital systems to be a significant component of power consumption, which is also the case here. If we operated the processor at a higher frequency, power consumption would increase considerably. In terms of plaintext sizes, overall average power consumption increases insignificantly (about 3%) with input size. Essentially, in the range from 1KB to 10KB of plaintext size which is of interest for our case, power consumption does not seem to be affected. This agrees also with the findings of Law et al. [19]. In accordance to the same work as well as our own measurements, a significant difference in consumed power would be observed in plaintext sizes comparable to the block size of the ciphers, i.e. 10 to 30 Bytes. In this range, key-initialization tasks place a computational overhead comparable to the actual encryption process. This indicates that encryption becomes more power-efficient with larger plaintext sizes.

A final observation from Fig. 1 is that the power-behavior of the ciphers does not change with increasing plaintext size, at least in the range of interest. There is one exception: 3-WAY and XXTEA switch places when moving to the larger plaintext but this is of minimal significance since they both score the poorest in terms of average power consumption. The best performing ciphers on this metric are IDEA, LOKI91, SKIPJACK, MISTY1 and RIJNDAEL. Although DES is included in the study as a reference point, it cannot be selected as a winning candidate in the profiling due to its compromised status. It is interesting, however, to observe that it features one of the lowest power profiles even though it is one of the oldest encryption algorithms.

Except for average power consumption, another interesting metric is peak power consumption. This is especially important for battery-powered systems such as implants are. A battery able to support a cipher with a given average power consumption may be unable to deliver the required output at a given point in time if the cipher sporadically presents peak power values which are largely deviating from its average power needs. To address this aspect of the profiled ciphers, we have plotted Fig. 2. The ciphers are depicted in order of increasing peak-power profiles. The bar series denoted as average power consumption is the aggregated equivalent of the bars seen previously, in Fig. 1.

It is interesting to see that ciphers scoring high in the previous test, such as IDEA and LOKI91, display a large dif-

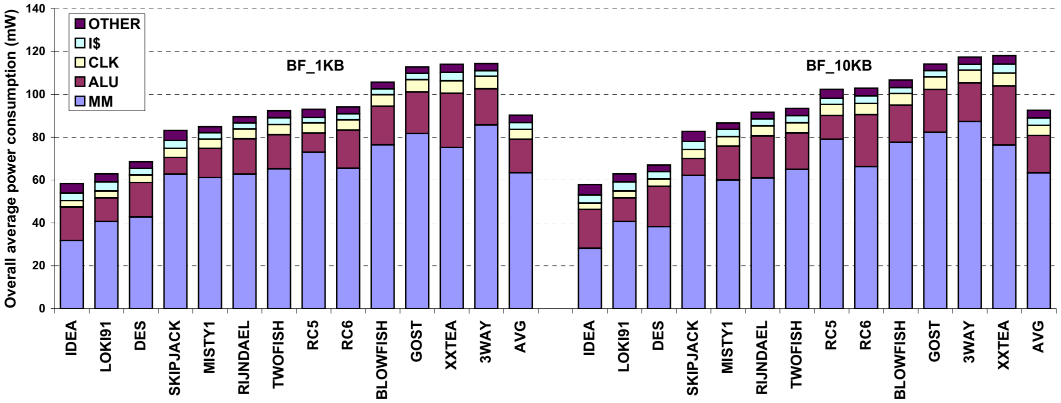


Figure 1: Per-component, average power consumption (in mW) for two plaintext sizes.

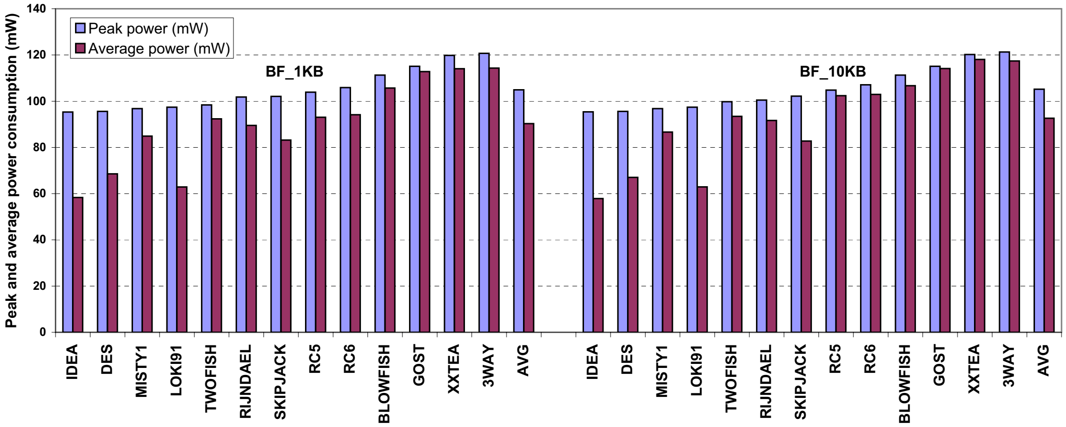


Figure 2: Average and peak power consumption (in mW) for two plaintext sizes.

ference of roughly 35 mW between average and peak power, which can potentially throw the implant designer off track. This difference has to be taken into serious account if such ciphers are to be employed in an implantable device. Having said that, IDEA, MISTY1, LOKI91 and RIJNDAEL still occupy the first positions. However, TWOFISH in now inside the top-scoring ciphers and, what is more, it displays the most consistent profile between average and peak power. In terms of our chosen plaintext sizes, and similarly to average power, peak-power profiles present no differences.

5.2 Energy expenditure

Apart from the rate at which a cipher consumes energy, i.e. its power consumption, it is important to also investigate the total energy costs incurred for executing the whole cipher. This metric is the total energy expenditure of a cipher and our findings are summarized in Fig. 3, for both plaintext sizes. SKIPJACK and LOKI91 have been omitted from the plots since they display excessively large energy needs (an order of magnitude larger for LOKI91 than the rest of the algorithms). However, average values include these two algorithms in their calculation to give a complete view.

Knowing the overall energy budget needed for completing a single encryption task is especially important for implantable systems. It directly tells us how much stored energy the given task needs in order to execute and, in effect, what energy amount will be deduced from the battery. It also tells us if e.g. a scheduled encryption and transmission of physiological readouts can take place or not. Given the mission-critical tasks implants perform, it might be preferable at some point to not engage in transmission of (encrypted) data. For instance, it is more important for a pacemaker running low on battery to keep working for an extra couple of days (to allow time for recharging or servicing) than to transmit ECG readouts to its inquiring host once and then power down.

In terms of energy distribution in the various processor components, we can again see that the MM, ALU, CLK and I$ are the most demanding ones. However, Fig. 3 tells a completely different story for the energy sparingness of the profiled ciphers. RC6 and RC5 have climbed in the first positions of the ranking, becoming the most energy-efficient ciphers. IDEA and MISTY1 follow with RIJNDAEL and BLOWFISH contesting for the fifth position across the two different plaintexts. Clearly, MISTY1 and RIJNDAEL perform better when smaller plaintext sizes are considered. Conversely, BLOWFISH favors larger sizes. Further, it is surprising that XXTEA is not among the best-scoring ciphers since it is considered a relatively light-weight algorithm.

A last observation in this subsection is that energy budget does not scale linearly with plaintext size for most of the

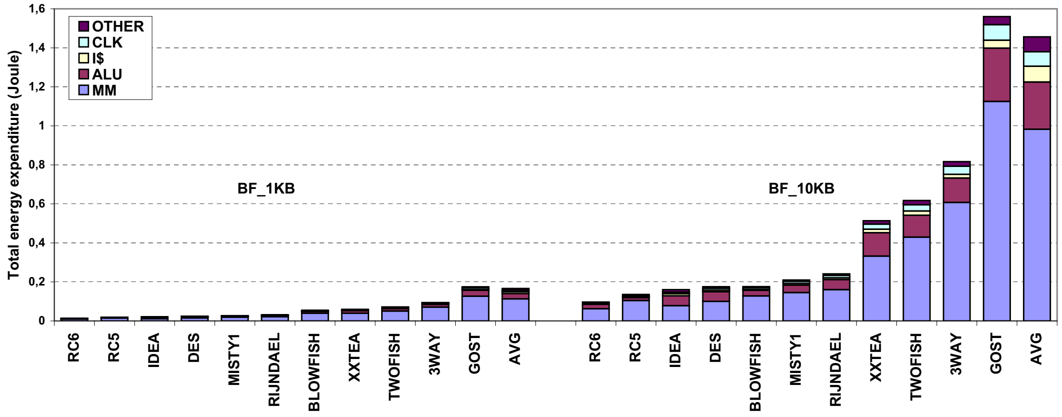


Figure 3: Per-component and total encryption energy costs (in Joules).

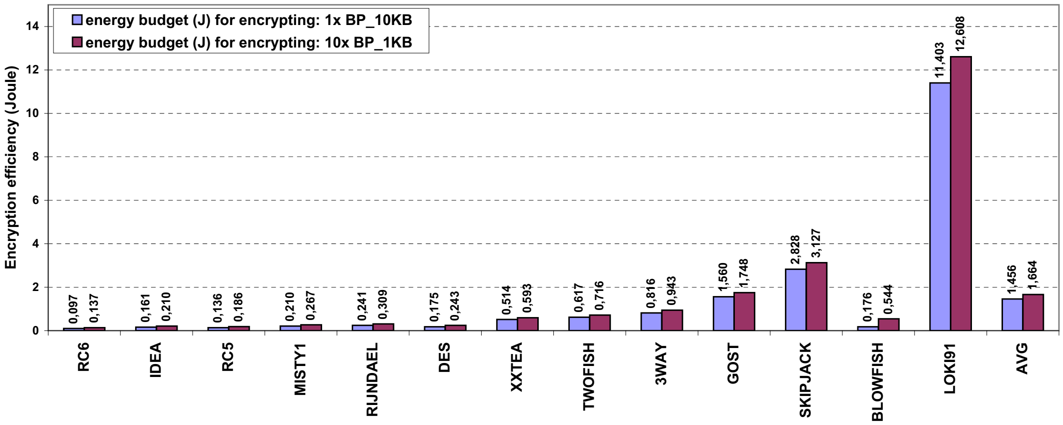


Figure 4: Computation overhead of ciphers manifested as energy penalty (in Joules) when encrypting one 10-KB and ten 1-KB plaintexts.

ciphers. The cost of encrypting a 10-KB workload as opposed to that of successively encrypting 10 1-KB workloads is 14% smaller, in an overall. The reason for that difference again is the overhead penalty paid during initialization of the encryption algorithms (e.g. key setup). As our simulations have revealed, other factors also contributing to this penalty are the increased fetch- and data-stalls that are reduced over the execution time of a cipher as cache entries get filled, etc.. However, this penalty is not similar across the various ciphers. In Fig. 4, the energy budgets for encrypting one 10-KB workload and 10 consecutive 1-KB workloads are plotted. The ciphers are ranked in order of increasing difference between the two budgets, i.e. in order of increasing penalty. RC6, IDEA, RC5, MISTY1 and RIJNDAEL are still in the first positions, incurring small penalties but TWOFISH has fallen near the bottom of the ranking, due to introducing a significant energy penalty. This secondary metric of energy is interesting because it indirectly gives a measure of computational efficiency of the various ciphers.

5.3 Encryption rate

Another metric we use in our profiling study of block ciphers is their encryption rate. In Fig. 5 encryption rates in KB/sec are reported for 1-KB and 10-KB plaintexts. RC6, RC5, MISTY1, RIJNDAEL and BLOWFISH score the highest on this metric, with RC6 and RC5 being by far the fastest ciphers. In fact, and contrary to the rest of the ciphers, RC6 and RC5 achieve impressive encryption-rate improvements with increasing plaintext size. The rate of BLOWFISH appears also to benefit largely from a larger plaintext size. In an overall, all ciphers seem to benefit from larger plaintexts: from 3.34 KB/sec for the 1-KB data, the average rate boosts to 4.52 KB/sec. The reasons for this are the same as the ones previously mentioned concerning the energy penalty. They are related to the cipher-key initialization phase as well as the cold start of the processor itself.

For the targeted implant applications, our primary concern is to preserve power consumption at low levels. This means that we are not seeking the fastest performing cipher but, rather, one which is fast enough to cover our needs. As can be seen in Table 1, the biological data we used as plaintext features a relatively high (in this context) sampling rate of 4.86 KB/sec for the 1-KB and 5.22 KB/sec for the 10-KB workload. In our 2-MHz simulated processor, only ciphers RC6 and RC5 manage to sustain the required sampling rate. The cost paid is that both ciphers display a relatively high power profile (93 mW to 100 mW) as seen in subsection 5.1.

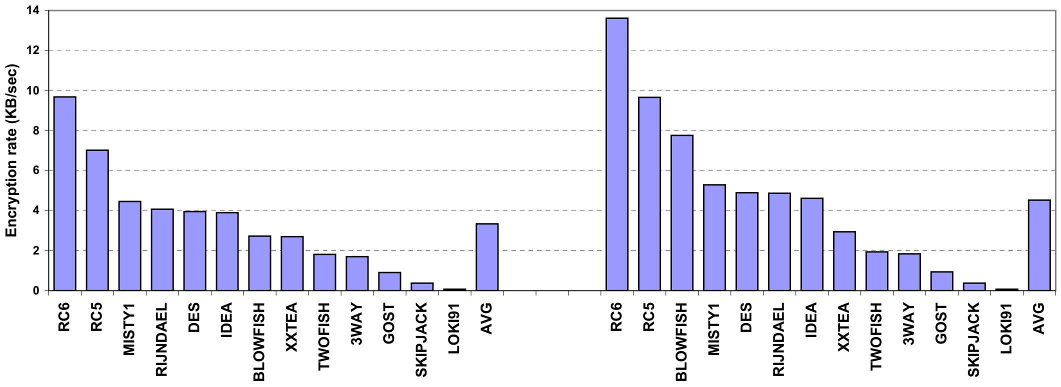


Figure 5: Encryption rate (in KB/sec).

5.4 Executable-binary size

In order to give a measure of proportion to our profiling study, it is useful to also report on the binary size of the encryption-algorithms executables, as a measure of program-memory needs. Since XScale supports the ARM ISA and, accordingly, XTREM is based on a modified version of SimpleScalar/ARM, executables have been built with the GNU ARM-GCC v4.1.2 cross-compiler. Furthermore, executables have been statically linked (this is an ARM requirement) and, therefore, are expected to be somewhat larger in size than their e.g. 8086-architecture version. Optimization level 2 (-O2 flag) has been used instead of level 3 (-O3 flag). It could possibly make faster code but the applications that benefit from it are very few, usually image and video decoders. However it has a side effect: it always generates a larger binary sizes. Since video/audio applications are not included in our workloads and we try to avoid large binaries as much as possible, O2 was selected as a proper optimization level. In Table 4, the code complexities of the selected encryption algorithms are shown in ascending order.

Obviously, results shown in the table are implementation-dependent and should be considered with caution. However, as we mentioned also in section 4.2, many different algorithms have been based on the same infrastructure (or suite), built by the same author(s). Therefore, the difference in sizes (not the actual sizes themselves), can give an indication of the difference in program-memory needs, regardless of the underlying implementations. Best scoring algorithms in this case are XXTEA, 3WAY, LOKI91, RC6 and RC5.

encryption algorithm	size (KB)
XXTEA	11.2
3WAY	11.2
LOKI91	11.3
RC6	11.4
RC5	11.4
GOST	12.2
SKIPJACK	12.2
IDEA	13.4
DES	14.6
BLOWFISH	15.3
MISTY1	18.8
TWOFISH	22.2
RIJNDAEL	37

Table 4: Program sizes (in KB) of the encryption algorithms.

5.5 Security margin

Since we are evaluating encryption ciphers, a last, suitable metric of our comparative study is the security level provided by each cipher. According to Lenstra and Verheul [20], a cryptosystem can be assumed to be secure only if it is considered to be sufficiently infeasible to mount a successful attack. Unfortunately, it is hard to quantify what precisely is meant by ”sufficiently infeasible”. To cope with this known issue, we adopt the widely used *security margin* metric, proposed also by Lenstra and Verheul, which is defined as the year until which a user was willing to trust the DES cipher.

According to this definition, if an attacker could afford C_{DES} computations in 1982, sufficient to break DES, and can afford C_X computations in year y ($y > 1982$), sufficient to break cipher X, then the security of cipher X in year y is computationally equivalent to the security of DES in 1982, or in other words, the security margin of cipher X is y. Since DES was standardized in 1977 and set for review in 1982, the year 1982 is used as the baseline. If the best known attack against a cipher with key length k is exhaustive key search, y can be calculated according to: $y = 1982 + \frac{30}{23} * (k - 56)$.

Security margins are shown in Table 5 in descending order. Based on the previous discussion, all algorithms except for LOKI91 and (of course) DES are secure. Also, SKIPJACK,

encryption algorithm	key size (bits)	Security margin
GOST	256	2243
BLOWFISH	128	2076
IDEA	128	2076
RC5	128	2076
XXTEA	128	2076
MISTY1	128	2076
RC6	128	2076
TWOFISH	128	2076
RIJNDAEL	128	2076
3WAY	96	2034
SKIPJACK	80	2013
LOKI91	64	1992
DES	56	1982

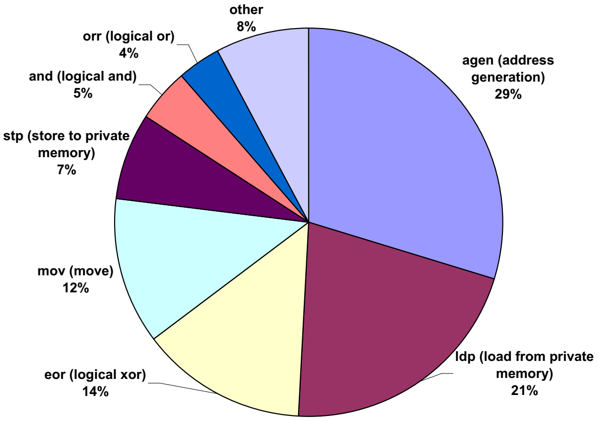
Table 5: Security margins of the encryption algorithms.

average power consumption	peak power consumption	total energy cost	encryption efficiency	encryption rate	program-code size
IDEA	IDEA	RC6	RC6	RC6	XXTEA
LOKI91	MISTY1	RC5	IDEA	RC5	3WAY
SKIPJACK	LOKI91	IDEA	RC5	MISTY1	LOKI91
MISTY1	TWOFISH	MISTY1	MISTY1	RIJNDAEL	RC6
RIJNDAEL	RIJNDAEL	BLOWFISH	RIJNDAEL	BLOWFISH	RC5

Table 6: Five best-performing encryption algorithms (in descending order of performance).

ARM instruction	Equiv. ARM microcode
stmdb r13!,r4-r8,r10-r15	agen tmp1,r13,0 agen tmp0,tmp1,-16 stp r11,[tmp0] agen r13,r13,-16 agen tmp0,tmp1,-12 stp r12,[tmp0] agen tmp0,tmp1,-8 stp r14,[tmp0] agen tmp0,tmp1,-4 stp r15,[tmp0]

(a) Sample ARM instruction which stores registers in the stack pointed to by R13 and equivalent ARM $\mu - OP$ sequence.



(b) u-op mix and frequencies for MISTY1 operating on the 1-KB BP plaintext.

Figure 6: ARM microcode and MISTY1 instruction frequencies

IDEA		RC6	
u-op	**percentage**	**u-op**	**percentage**
mov (move)	29%	**agen (address generation)**	25%
agen (address generation)	18%	**mov (move)**	15%
ldp (load from private memory)	12%	**ldp (load from private memory)**	15%
b (unconditional branch)	9%	**stp (store to private memory)**	8%
add (add)	7%	**add (add)**	8%
cmp (compare)	6%	**b (unconditional branch)**	5%
stp (store to private memory)	5%	**eor (logical xor)**	4%
orr (logical or)	4%	**sub (subtract)**	4%
other	10%	**rsb (reverse subtract)**	4%
		other	11%

Table 7: u-op mix and frequencies for IDEA and RC6 operating on the 1-KB BP plaintext.

although secure, displays the next shortest security margin. Conclusively, LOKI91 and SKIPJACK - if it does not appear to rank high in the rest of the metrics - will be dropped from our final selection process.

5.6 Cipher selection and results

To summarize our analysis results, we present in Table 6 the 5 best-performing algorithms in each one of our profiled metrics (except for the security-margin metric). MISTY1 appears in 5 out of 6 metrics in the above table. IDEA, RIJNDAEL, RC6 and RC5 follow with each with 4 occurrences in the table. However, IDEA performs consistently better than RIJNDAEL with the exception of encryption rate. Besides, RIJNDAEL scores almost always last in the ranking among ciphers with 4 occurrences. Last, RC6 scores always better than RC5. LOKI91 has 3 occurrences in the table but is, in any case, dismissed due to its now insecure nature.

Conclusively, from the above findings, MISTY1 is the most promising cipher according to our imposed metrics; thus, we take a closer look at its underlying instruction mix. Fig. 6(b) illustrates the type and frequency of instructions executed for encrypting the 1-KB BP plaintext with the MISTY1 cipher. XTREM, which is based on SimpleScalar, implements ARM instructions through microcode (referred to as $\mu - ops$ hereon). We included $\mu - op$ statistics rather than ARM instruction because they can better capture the workings of the underlying architecture. For instance, a single ARM command to store multiple registers to the stack pointed to by R13, breaks down to a number of more elementary $\mu - ops$ (see Fig. 6(a)).

Going back to Fig. 6(b), we readily observe that execution is heavily dominated by load/store (stp, ldp) operations, logic operations (eor, and, orr) and register-to-register copy operations (mov). Such a mix also explains the dominance of the MM and ALU components in the power-consumption plots, as previously discussed. This mix motivates us towards efficient implementation of loads/stores, moves and logic operations in terms of power consumption and speed. Note that ”agen” is always used to calculate an address before load/store-related operations and, thus, is not considered as a stand-alone instruction but as part of those operations.

By investigating also the second and the third best ciphers, i.e. IDEA and RC6, we accumulate the following statistics, seen in Table 7. The mixes in this case favor load/store and move operations highly but logic operations to a smaller extent, compared to the MISTY1 case. However, they both display high percentages of arithmetic (add, sub, rsb, cmp) and branch (b) operations, contrary to MISTY1. Given that MISTY1 scores high in most metrics of our profiling study, optimizing our architecture for the more focused MISTY1 instruction mix alone is considered the best option.

6. CONCLUSIONS

In this work we have examined a number of metrics, in the context of biomedical microelectronic implants, of various symmetric encryption algorithms on a workload suite of recorded biological signals. The value of this research lies not in specifying absolute performance values for the given ciphers nor in detailing their specific workings. Rather, it offers insight on the relative behavior of the ciphers operating on different workload sizes as well as on qualitative results regarding the trade-offs among different metrics such as encryption rate and average power consumption.

In view of designing the architecture of a novel microprocessor for biomedical implants, the ulterior goal of this work is to identify the most common instructions executed in the best-performing cipher of our benchmark suite. The winner has been found to be MISTY1 and, by analyzing it, we found that load/store, reg-to-reg move and logic operations are the dominating ones in its execution. It is these instructions that we will try to address most efficiently in our architecture.

Future work for this study includes extending our benchmark suite with different classes of applications, other than symmetric cryptography. Classes we are currently considering are: i) communication-specific applications, to profile over-the-air transmitted data, ii) lossless data-compression algorithms, to profile compact storage and over-the-air transmitted data, and iii) applications (or application segments) found in actual, working implantable systems; in our case, software running in (non-) commercial biomedical implantable devices.

7. ACKNOWLEDGEMENTS

This work has been partially supported by the ICT Delft Research Centre (DRC-ICT) of the Delft University of Technology. It would not have been complete without the valuable contributions of Prof. Stefanos Kaxiras and Gilberto Contreras for providing an excellent power simulation tool and support. Also, many thanks are due to Christoforos Kachris, Carlo Galuzzi, Pepijn de Langen, Filipa Duarte and Vlad-Mihai Sima for their valuable comments and help throughout.

8. REFERENCES

[1] 3-WAY, BLOWFISH, DES, GOST, IDEA, RC5 source code. `www.cis.udel.edu/~mills/database/schneier/`.

[2] SKIPJACK, LOKI91 source code. `www.mirrors.wiretapped.net/security/cryptography/algorithms/skipjack/`.

[3] University of Michigan, Sim-Panalyzer 2.0. `http://www.eecs.umich.edu/~panalyzer/`.

[4] Austin, T., Larson, E., and Ernst, D. SimpleScalar: an infrastructure for computer system modeling. *IEEE Computer 35*, 2 (February 2002), p. 59–67.

[5] Berkman, J., and Prak, J. Biomedical microprocessor with analog I/O. In *IEEE International Solid-State Circuits Conference - Digest of Technical Papers* (19 February 1981), p. 168–169.

[6] Brooks, D., Tiwari, V., and Martonosi, M. Wattch: A Framework for Architectural-Level Power Analysis and Optimizations. In *ISCA'00* (2000), p. 83–94.

[7] Burger, D., and Austin, T. The SimpleScalar tool set, version 2.0. *ACM SIGARCH Computer Architecture News 25*, 3 (June 1997), p. 13–25.

[8] Chang, C.-C., Muftic, S., and Nagel, D. Measurement of energy costs of security in wireless sensor nodes. In *16th International Conference on Computer Communications and Networks (ICCCN)* (2007), p. 95–102.

[9] Contreras, G., and Martonosi, M. The XTREM Power and Performance Simulator for the Intel XScale Core: Design and Experiences. *ACM Transactions on Embedded Computing Systems 6*, 1 (February 2007), p. 1–25.

[10] Contreras, G., Martonosi, M., Peng, J., Ju, R., and Lueh, G.-Y. XTREM: A Power Simulator for the Intel XScale Core. In *LCTES'04* (2004), p. 115–125.

[11] Du, W., Deng, J., Han, Y., Varshney, P., Katz, J., and Khalili, A. A pairwise key predistribution scheme for wireless sensor networks. *ACM Transactions on Information and System Security (TISSEC) 8* (May 2005), p. 228–258.

[12] Ector, H., and Vardas, P. Current use of pacemakers, implantable cardioverter defibrillators, and resynchronization devices: data from the registry of the european heart rhythm association. *European Heart Journal Supplements 9* (August 1988), p. 144–149.

[13] Flick, B., and Orglmeister, R. A portable microsystem-based telemetric pressure and temperature measurement unit. In *IEEE Transactions on Biomedical Engineering* (Jan. 2000), vol. 47, p. 12–16.

[14] Gaubatz, G., Kaps, J.-P., and Sunar, B. Public key cryptography in sensor networks - revisited. In *Lecture Notes in Computer Science* (2005), p. 2–18.

[15] Grossschadl, J., Tillich, S., Rechberger, C., Hofmann, M., and Medwed, M. Energy evaluation of software implementations of block ciphers under memory constraints. In *Conference on Design, automation and test in Europe* (2007), p. 1110–1115.

[16] Harrigal, C., and Walters, R. The development of a microprocessor controlled implantable device. In *IEEE Proceedings of the 1990 Sixteenth Annual Northeast Bioengineering Conference* (Mar. 1990), p. 137–138.

[17] Intel Corporation. *Intel XScale Microarchitecture for the PXA255 Processor: User's Manual*, March 2003.

[18] Jeong, Y.-S., and Lee, S.-H. Hybrid key establishment protocol based on ecc for wireless sensor network. *Lecture Notes in Computer Science 4611* (August 2007), p. 1233–1242.

[19] Law, Y., Dourmen, J., and Hartel, P. Survey and benchmark of block ciphers for wireless sensor networks. *ACM Transactions on Sensor Networks 2* (February 2006), p. 65–93.

[20] Lenstra, A., and Verheul, E. Selecting cryptographic key sizes. *Journal of Cryptology: the journal of the International Association for Cryptologic Research 14*, 4 (2001), p. 255–293.

[21] Luo, X., Zheng, K., Pan, Y., and Wu, Z. Encryption algorithms comparisons for wireless networked sensors. In *IEEE International Conference on Systems, Man and Cybernetics* (2004), p. 1142–1146.

[22] Min, M., Parve, T., Kukk, V., and Kuhlberg, A. An implantable analyzer of bio-impedance dynamics - mixed signal approach. In *IEEE Instrumentation and Measurement* (Budapest, Hungary, 21-23 May 2001), p. 38–43.

[23] Sawan, M., Robin, S., Provost, B., Eid, Y., and Arabi, K. A wireless implantable electrical stimulator based on two FPGAs. In *Proceedings of the IEEE International Conference on Electronic Circuits and Systems (ICECS)* (Piscataway, New Jersey, USA, 1996), vol. 2, p. 1092–1095.

[24] Schwarz, M., Ewe, L., Hauschild, R., Hosticka, B., Huppertz, J., Kolnsberg, S., Mokwa, W., and Trieu, H. Single chip CMOS imagers and flexible microelectronic stimulators for a retina implant system. In *Sensors and Actuators A: Physical* (22 May 2000), vol. 83, p. 40–46.

[25] Shults, M., Rhodes, R., Updike, S., Gilligan, B., and Reining, W. A telemetry-instrumentation system for monitoring multiple subcutaneously implanted glucose sensors. In *IEEE Transactions on Biomedical Engineering* (Oct. 1994), vol. 41, p. 937–942.

[26] Smith, B., Tang, Z., Johnson, M., Pourmehdi, S., Gazdik, M., Buckett, J., and Peckham, P. An externally powered, multichannel, implantable stimulator-telemeter for control of paralyzed muscle. In *IEEE Transactions on Biomedical Engineering* (1998), vol. 45, p. 463–475.

[27] Strydis, C., Gaydadjiev, G., and Vassiliadis, S. Implantable microelectronic devices: A comprehensive review. CE-TR-2006-01, Computer Engineering, Delft University of Technology, December 2006.

[28] Valdastri, P., Menciassi, A., Arena, A., Caccamo, C., and Dario, P. An implantable telemetry platform system for in vivo monitoring of physiological parameters. In *IEEE Transactions on Information Technology in Biomedicine* (Sept. 2004), vol. 8, p. 271–278.

[29] Venugopalan, R., Ganesan, P., Peddabachagari, P., Dean, A., Mueller, F., and Sichitiu, M. Encryption overhead in embedded systems and sensor network nodes: modeling and analysis. In *International Conference on Compilers, Architecture and Synthesis for Embedded Systems* (2003), p. 188–197.

[30] Wander, A., Gura, N., Eberle, H., Gupta, V., and Shantz, S. Energy analysis of public-key cryptography for wireless sensor networks. In *3rd IEEE International Conference on Pervasive Computing and Communications* (2005), p. 324–328.

[31] Warren, J., Dreher, R., Jaworski, R., Putzke, J., and Russie, R. Implantable cardioverter defibrillators. In *Proceedings of the IEEE* (1996), vol. 84, p. 468–479.

[32] Wheeler, D., and Needham, R. Correction to XTEA. Tech. rep., Computer Laboratory, University of Cambridge, October 1998.

[33] Wouters, P., Cooman, M. D., Lapadatu, D., and Puers, R. A low power multi-sensor interface for injectable microprocessor-based animal monitoring system. In *Sensors and Actuators A: Physical* (1994), vol. 41-42, p. 198–206.